Fault Management Guiding Principles

Marilyn E. Newhouse[1]
CSC, Marshall Space Flight Center, Alabama, 35812, USA

Kenneth H Friberg[2] and Lorraine Fesq[3]
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA,91109, USA

and

Bryan Barley[4]
National Aeronautics and Space Administration, George C. Marshall Space Flight Center, Marshall Space Flight Center, Alabama, 35812, USA

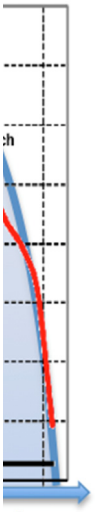
Regardless of the mission type: deep space or low Earth orbit, robotic or human spaceflight, Fault Management (FM) is a critical aspect of NASA space missions. As the complexity of space missions grows, the complexity of supporting FM systems increase in turn. Data on recent NASA missions show that development of FM capabilities is a common driver for significant cost overruns late in the project development cycle. Efforts to understand the drivers behind these cost overruns, spearheaded by NASA's Science Mission Directorate (SMD), indicate that they are primarily caused by the growing complexity of FM systems and the lack of maturity of FM as an engineering discipline. NASA can and does develop FM systems that effectively protect mission functionality and assets. The cost growth results from a lack of FM planning and emphasis by project management, as well the maturity of FM as an engineering discipline, which lags behind the maturity of other engineering disciplines. As a step towards controlling the cost growth associated with FM development, SMD has commissioned a multi-institution team to develop a practitioner's handbook representing best practices for the end-to-end processes involved in engineering FM systems. While currently concentrating primarily on FM for science missions, the expectation is that this handbook will grow into a NASA-wide handbook, serving as a companion to the NASA Systems Engineering Handbook. This paper presents a snapshot of the principles that have been identified to guide FM development from cradle to grave. The principles range from considerations for integrating FM into the project and SE organizational structure, the relationship between FM designs and mission risk, and the use of the various tools of FM (e.g., redundancy) to meet the FM goal of protecting mission functionality and assets.

I. Introduction

In April 2008, the National Aeronautics and Space Administration's (NASA) Science Mission Directorate (SMD) Planetary Science Division (PSD) sponsored the first in what is hoped will become a series of Fault Management Workshops. The workshop was initiated as a response to the cost overruns in FM development and testing experienced in a number of recent PSD missions, and identified as a significant driver for total life cycle cost overruns in a study performed by the Discovery and New Frontiers Program Office at Marshall Space Flight Center[1]. FM cost overruns for a representative project are illustrated in Figure 1. The workshop was conceived as the first step in understanding and ultimately controlling these cost overruns.

[1] Principal Lead Systems Engineer; MSFC/VP23.
[2] Chief Engineer, Friberg Autonomy, LLC Portland, OR, AIAA Senior Member
[3] Principal Engineer, Engineering Development Office, Systems and Software Division, AIAA Senior Member.
[4] Lunar Quest Program Chief Engineer, MSFC/EE04.

One of the recommendations[2] from the workshop was to capture the current knowledge and best practices regarding FM in a handbook, available to practitioners as a training and education tool. To this end, the NASA's SMD and the NASA Engineering Safety Council (NESC) provided funding for a multi-institution team to develop the preliminary version of a NASA Fault Management Handbook[3] (hereafter, referred to as the NASA FM Handbook). The handbook is intended to be a companion to the NASA Systems Engineering Handbook[4], and ultimately it is intended to address FM systems across the full spectrum of NASA programs and flight projects: aeronautics, human spaceflight, and robotic space missions. The goals of the NASA FM Handbook are to:



ve plot of
the NASA

- Promote recognition of FM as an engineering discipline
- Expound and establish foundational FM concepts and guiding principles
- Raise awareness of FM recommended practices
- Note institutional and programmatic factors that substantially affect FM
- Promote organizational structures that facilitate effective FM development
- Delineate a FM development process and lifecycle consistent with the NASA SE Handbook
- Articulate the purpose, process, work products, potential pitfalls, and recommended practices across the FM lifecycle
- Allow projects to avoid repeated FM lessons learned

Organized by life cycle phase, the NASA FM Handbook discusses principles, best practices and highlights pitfalls that FM practitioner's have encountered throughout the FM development life cycle. It also includes a subset of lessons learned specific to FM mined from the NASA Lessons Learned database.

Due to time and funding constraints, the preliminary version of the FM Handbook primarily addresses robotic space missions. It pulls together the collected wisdom across NASA robotic programs and projects regarding the life cycle processes, specifically as they address the formulation, implementation, and operation of FM systems. This paper provides a snapshot of the guiding FM principles included in the preliminary version of the FM Handbook that drive the detailed organizational, architecture, design, implementation, test, and operations pitfalls, lessons learned, and best practices for robotic space missions. At the time of this paper, the NASA FM Handbook is being distributed to the NASA centers for formal review and will be revised as necessary based on the comments received from that review.

II. Failures, Faults, and Anomalies

Discussions of FM are fraught with confusion resulting from differences in terminology. In the FM Handbook, "failure" is defined as the unacceptable performance of an intended function, but not necessarily the loss of all functionality. "Fault" is then defined as the internal cause of the failure. In this sense, failure is something to be detected, fault is something to be determined, and the two are linked by cause (fault) and effect (failure).

Other engineers use fault to describe the detected occurrence of undesirable performance. In this sense, a fault is an event to be explained. The fault may be the result of a failure, it may be an indicator of a potential future failure, or it may simply be an anomaly that does not materially affect system performance. Because in this case the fault is the item detected and a failure may be a cause of the fault, this usage is diametrically opposed to the previous definitions.

Adding to the confusion, When faults and failures are linked by cause and effect, the linkage is hierarchical, as shown in Figure 2. A fault determined to be the cause of a failure can be seen from another perspective as a failure caused by a deeper, underlying fault. A well-known example of this is the *Columbia* shuttle accident.[1] In the initial

investigation into the cause(s) of the accident, the event to be explained was the breakup of the orbiter. This was soon explained by the weakening of the wing due to overheating, traced to a deeper cause, the external tank foam falling off during ascent and punching a hole in the wing's leading edge Reinforced Carbon-Carbon. For many, this was "the physical explanation," or "the physical cause" of the accident. For external tank designers, and for the Columbia Accident Investigation Board, this cause was an effect that needed to be explained. Thus, in discussions, what one engineer refers to as a fault (cause) can be seen by another as a failure (effect).

A common understanding of terminology is essential for accurate communication. Requirements are often written with contradictory uses of the terms fault and failure. For the unwary, this can lead to different interpretations of requirements, and to latent faults in the design that in turn can result in catastrophic failure of the system in operations. Each project will need to come to an agreement regarding how fault and failure are to be defined and used within the team, and maintain an awareness that the terms are often used differently and that the differences can result in misunderstanding.

fault
n one
nother

Finally, the term "anomaly" is often used in connection with FM. In the FM Handbook, an anomaly is defined as the unexpected performance of an intended function. Although an anomaly can be a failure, it should not be confused with a failure. Failures can exist without being anomalous (e.g., the expected depletion of an expendable resource), and anomalies that are not failures are also common (e.g., an unusual (unexpected) power signature that does not cause any loss of functionality).

Failures, not anomalies, are the primary focus of FM. However, the FM practitioner should consider the potential for anomalies as well as possible failures. Anomalies can be used as predictors for future faults, as in the case of an increase in temperature that is within the normal operating range but approaches the limiting value. Anomalies in one area can also lead to faults in other areas, as in the case of an increase in temperature in a component that causes overheating and failure of a neighboring component.

III. Principles

Principles are basic truths, generalizations, laws, or assumptions that are accepted as true and that can be used as a basis for reasoning or conduct. For the preliminary version of the NASA FM handbook coming up with a consistent, agreed upon, succinct list of salient principles was challenging partly because many institutions haven't formalized what their FM principles are – more often than not, it is a reversed engineered list from distilling the high level principles out of good FM engineering practices, and it can be a gray area determining the line between a concept, a practice or a principle. That said below are eight good FM principles identified in the FM Handbook, though it is expected that future revisions may identify additional candidates.

A. FM as a Cross-Cutting Engineering Discipline

Principle: FM is a crosscutting engineering discipline that requires close coordination with systems engineering, Safety & Mission Assurance (S&MA) / Reliability, and subsystem engineering teams.

FM is often allocated as a subset of the responsibilities of the systems engineers. However, FM is its own discipline that requires independent resources to ensure that the appropriate level of attention is given to FM throughout the full project life cycle. FM must be allocated, designed, analyzed, verified, and validated in ways that cross specific implementation areas. This means that FM should be organized as a set of system tasks and functions, and not merely in a disciplinary or subsystem fashion. The FM engineers require visibility into the functionality of the entire system to identify and plan appropriate responses to off-nominal behaviors; they need to ensure that FM is

considered during system trade studies and often must force trades at various levels and across multiple subsystems or within the subsystems. FM requirements must be developed at the system level to ensure a cohesive approach. FM engineering utilizes and contributes to the results of traditional reliability analyses. Implementation of FM functions typically is distributed across all elements of the project: hardware, software, and operations. Thus, FM requirements must be defined and clearly allocated to the implementing teams early enough to ensure that the effort is fully estimated and the required implementation resources are available.

However, unlike systems engineering, FM is also a subsystem with flight and ground system deliverables. Even if most of the implementation is allocated outside of the FM team (e.g., to the FSW team), the FM engineers are responsible for FM requirements development and allocation, FM analysis, FM algorithm and parameter development, FM system and subsystem testing, and FM testing and operations procedures.

Therefore, a project's organizational structure and delegation of roles/responsibilities/authority must support the flow of information to and from FM engineering, and allow trades to be clearly communicated and resolved across traditional subsystem and engineering disciplines. FM engineers need to be constantly aware of the global nature of engineering decisions that can affect FM and FM decisions that can affect overall system complexity and operations. FM engineers need to be aware of and coordinate with the scheduled activities of the various project teams.

B. FM Scope and Boundary

Principle: *Specify the system boundary so that it encompasses everything that detects, evaluates, and responds to failures as part of the system, including vehicle, crew, operators, and ground systems. The environment typically lies outside of the boundary; however, the system must function within expected environmental conditions.*

The placement of the system boundary is an essential concept for FM. The system boundary defines the limit of responsibility and/or interest, beyond which the team or engineer is not required to control faults. The FM boundary also clearly identifies the full set of functionality encompassed by FM.

Outside of the boundary lies the environment, which the system cannot alter, but within which the system must execute its mission. Although the environment lies outside of the system, the FM practitioner must understand the interactions of the system across the boundary to the environment to ensure the system functions properly within the environment. As an example, the expected radiation levels through the life of the mission sets the environment within which the FM system must protect mission functionality. However, it may be beyond the mission resources and FM scope to attempt to preserve functionality through radiation levels resulting from the solar storm of the century. In either case, the careful specification of the system boundary, including the expected environmental conditions ensures the necessary FM protections are developed, while controlling unnecessary growth in capability and complexity.

Inside the boundary, FM functionality is typically distributed across multiple elements of the system and multiple phases of use, with specific (and often redundant) capabilities assigned to hardware, software, and operational elements. Depending on the mission design, risk posture, and resources available to the mission it can be common for the mission operators and/or crew (for human spaceflight missions) to perform essential FM functions. All hardware, software, procedures, and personnel that are required for implementing, testing, and operating the mission must be included within the FM boundary of the system.

There is also a "nominal operations" side within the FM boundary that must be addressed: setting FM parameters, developing spacecraft deployment sequences, monitoring FM processing, reporting on FM actions, and supporting troubleshooting of both system and FM behaviors. The FM design must ensure that the information required to trace and resolve faults or failures is available in telemetry and preserved through a cascade of faults/failures in order to allow ground reconstruction and root cause analysis.

The system-level FM engineer must address the entire FM scope, and must set the system boundary to encompass all mechanisms that perform FM functions. Once defined, the FM designer must carefully document the system boundary conditions that define the environment within which the system must correctly execute its function(s). These boundary conditions not only define the physical environment (e.g., thermal, radiation, wind, landing surface), but the risk posture accepted for each mission, and the operating environment (e.g., time delays necessitating autonomous operations) within which the mission must execute. This documented system boundary underpins the FM requirements and design, and helps control cost growth late in the development cycle.

C. FM Development as Part of Systems Engineering

Principle: *Design, analyze, verify, and validate FM with respect to the system's failure modes in parallel with development of the nominal system behaviors.*

Figure 3 is an illustration of the full functionality of every system, identifying both the "dark" side of potential failures and a "light side" of expected, nominal behaviors. The system's failure space is the set of possible failure behaviors, most of which will never occur in operation of the real system. Given the potential breadth of FM trades, decisions are often implicitly or explicitly made to postpone development of FM operational concepts, requirements, and designs, until the nominal side of system functionality is fairly mature. However, waiting to define and understand the potential failures limits the trades available when the analysis is finally performed and can lead to more expensive, complex, or risky solutions. To help control the complexity and ensure that the FM design is "dyed in" rather than "painted on," design and implementation of FM capabilities needs to progress hand-in-hand with the functions FM is expected to preserve.

ng the
ominal

D. Function Preservation

Principle: *Design FM to protect system functions when the risks of failure of that function are unacceptable. FM may be defined independently from known specific failure causes that can affect those functions.*

Where the risks of failure for a function are unacceptable, FM is deployed to preserve or recover that function, or to select a new goal that does not require the failed function. To do this, identify functions that support mission goals and analyze those functions to determine if the risk of failure of this function, given the system design for that function, is consistent with the project's defined risk posture. FM should be deployed 0to improve the dependability of that function or to change the goal to an acceptable, achievable objective.

Most FM mechanisms are applied to mitigate against explicit, known failure causes. However, FM should be designed not only from the bottom up based on predicted failure modes (frequently identified in the Failure Modes and Effects Analysis (FMEAs)). A bottom-up design will often result in a complicated, incomplete, and potentially fragmented FM design. The FM design must also account for incomplete human understanding of the system's failure behavior, for potentially large uncertainties in probabilistic estimates, and for failures of complex systems even when, or particularly when, these uncertainties have not been estimated. Therefore, the FM design should also be developed top-down based on an assessment of goals, objectives, and functions.

Humans can and do create systems beyond their full capability to understand. Aerospace systems exhibit complexity well beyond the capabilities of full human understanding, due to their disciplinary depth, large number of components, heterogeneity, and behavioral interactivity. It is impossible to know if all possible failure modes have been identified for systems of even moderate complexity. The inherent incompleteness of knowledge implies that that some FM functions must be deployed to protect system functions, independent of known specific failure causes that can affect those functions. These act as a "safety net" against non-predicted causes.

E. Asset Preservation

Statement of Principle: *Design and operate FM to preserve system assets when the risks of loss of that asset are unacceptable with respect to the goals of the mission. As with preservation of functions, FM may be defined independently from known specific failure causes that can affect the system mechanisms and assets.*

This principle is a corollary to, or sub-principle of function preservation, but is important enough to call out separately. For the system to achieve its goals and objectives, it must perform required functions, and in turn, these required functions are assigned to specific assets. Assets include not only the hardware, software, and people, but also entities such as power and consumables such as propellant. In general, to preserve system function, one must preserve its assets. To determine the proper strategy for preserving assets, the FM practitioner must refer back to the

system's overall goals and objectives, the mission's risk posture, and the functions that must be performed to achieve them.

For example, it is appropriate in many emergencies for the system to abandon some of its current functions to preserve assets for the long run. Spacecraft safing is the most common example. It is acceptable to abandon some current functions while preserving those functions that protect the vehicle and its assets by shedding loads, stopping the current mission activity, reducing functions to the very smallest and simplest set to enable pointing back to Earth so that mission operators can diagnose the fault and recover from the failure. This can be done because those functions can be interrupted in order to preserve assets for future use, when they are needed in the science-gathering phase of the mission. The functions are re-started, usually by ground-based operators, upon failure recovery and used at the crucial mission time.

F. Risk Reduction

Statement of Principle: *The FM implementation should always increase the reliability and safety of a system.*

FM is a tool to reduce and manage overall mission risk. As such FM should deploy highly reliable and effective mechanisms that can be shown to reduce the overall mission risk, even though FM inherently adds more physical and logical mechanisms and hence potentially more failure modes and paths.

In the zeal to preserve functionality and assets, it is easy for the FM practitioner to be caught in a spiral of trying to protect the protection. Even the most simplistic case, where in the process of detecting and responding to a fault the FM design introduces an alternate fault path, the FM practitioner may be doing nothing more than increasing the overall complexity of the system. Each FM detection/response should be carefully evaluated to ensure it does not increase the risk posture of the mission, and that the benefit of the preservation of function or assets outweighs the increase in system complexity. Even if FM can be designed to cover all failure modes per requirements, if there are inadequate verification and validation resources to ensure the increased complexity of the design is adequately tested, then either the FM scope or the test assets must be reconsidered.

G. FM Mechanism Allocation

Statement of Principle: *Allocate FM functions to the appropriate design mechanism types, including hardware, software, operations, or any combination thereof, keeping in mind the complexity of the evolving FM system and the risk posture and resource constraints for the mission.*

FM is often perceived as primarily a software function. For others, redundant hardware components are the cornerstone of FM. It is easy to concentrate on one type of failure, e.g., random part failure, or one FM strategy, e.g., design-time fault avoidance, to the exclusion of all else. Alternatively, FM designs often are inherited from previous missions without consideration of the applicability of the heritage FM capabilities and mechanisms to the current mission (from FM concept and architecture through operations). However, FM is not "one size fits all." The FM implementation for a 15-year flagship mission or a deep space mission with long return time delays will be more complex than that for a 1-year Explorer class mission in low Earth orbit. A simple mission with a single-string hardware design may require more onboard automation to meet mission goals and, therefore, a more complex software design, that a larger mission with significant hardware redundancy. FM requirements for human space flight are more extensive than for robotic "proof of concept" missions.

There are five strategies used by FM:

- failure prevention:
  - design-time fault avoidance and
  - operational failure avoidance,
- and, failure tolerance:
  - failure masking,
  - failure recovery, and
  - goal change,

along with a full spectrum of mechanisms that should be considered for implementing the FM strategies. Different FM strategies and mechanisms are appropriate for different failure modes and mission types or for different mission phases (i.e., design vs. implementation and operations).

In failure prevention, actions are taken to ensure that failures will not occur. Failures can be prevented by designing function and FM capabilities to minimize the risk of a fault and resulting failure. For example, common examples of design-time fault avoidance are the use of stricter quality assurance processes or higher quality parts, or applying increased mission margins. Failures also can be prevented operationally, when analysis is used to predict that a failure is likely occur in the future. Operations personnel can then take action, possibly reducing the frequency

of use of a component. For example, operations may change the spacecraft attitude profile to reduce momentum build-up and the use of thrusters for momentum dumps. A decision might be made to switch to a backup component to prevent a failure from interrupting a critical activity. Alternatively, a model parameter could be modified (e.g., increasing a thermal limit or a wheel spin-down time) to reflect changes to aging components or changes in the mission environment during different operational phases.

In failure tolerance, failures are allowed to occur, but their effects are mitigated or accepted. Failures can be masked, steps can be taken recover from a temporary failure before the failure compromises a mission goal, or as a final response mission goals can be changed to new, usually degraded goals, that can be achieved. Failure masking is a variant of failure response in which failure effects are "hidden" from the rest of the system. The most common example of failure masking is a voting scheme in which a failed component is outvoted by two other identical components.

Failure recovery is defined as the actions taken to return the system to operations after a failure. In some cases, operations after recovery may be identical to operations prior to the failure, with no change of goals or functions. This would be the case for failover to an identical redundant hardware component or a computer reboot. However, recovery to normal operation may require a new goal (one different from the original goal) for the system. An example of this would be turning off instruments to continue operations in a lower power configuration. Failure recovery can be an autonomous recovery by the flight system for sophisticated FM systems or may require intervention by the ground if time constraints allow. However, failure recovery may also include maintenance or supportability actions as a part of the failure recovery. An example is a launch vehicle scrub. The failure recovery in this case may include repair and/or replacement of the failed component, reloading propellant tanks, and recycling the launch sequence to a point where it can be restarted.

Goal change is defined as an action that alters the system's current goals. Goal changes occur for many reasons, not just for FM. It is therefore not exclusively an FM function, but is shared with many other vehicle and mission functions and capabilities such as mission planning and operations, operational modes, and vehicle configuration controls. However, the most typical FM goal change is "safing." Usually the goal change is to a "degraded goal" or a subset of the system's original goals. For example, with spacecraft safing, the current science objectives may be abandoned while the spacecraft maintains the goals of ensuring a power-positive system and a communications link with Earth. In the case of a human-rated launch vehicle, an ascent abort abandons the goal of achieving orbit, but maintains the goal of keeping the crew safe. To do this, it specifies a different, achievable goal -- to return the crew capsule and crew back to Earth.

Part of the consideration of the appropriate FM mechanism is timing. FM is effective only if its responses execute fast enough to mitigate the effect of the failures to which each FM response applies. The race condition between the latencies of the mechanisms for detection and response to a failure and the temporal evolution of failure effects as they propagate through the system must be assessed for every FM mechanism that is included in the system. The assessment must include all latencies including communication with the ground, required analysis and human response times.

Selection of FM strategies, and ultimately the FM mechanisms used to implement the strategies, needs to be driven by the required mission functionality, the available mission resources across the mission life cycle, and the accepted risk posture for the mission. In general, as with many other areas, the FM practitioner should select the simplest solution (mechanism) that provides the required protection to preserve mission functionality and assets within the mission context and constraints.

H. Tailoring Redundancy

Statement of Principle: *Mission attributes drive the use of redundancy.*

Redundancy is a fundamental aspect of FM designs. In fact, redundancy, particularly hardware redundancy, is often seen as the primary approach for preserving mission functionality. However, redundancy takes different forms based on the potential type of fault. There are four different approaches to redundancy: hardware identical, functional, informational, and temporal. Each of these approaches is better suited to handling different types of failures (e.g., common-mode/design faults, random part failure, or human error). When redundancy is included in the FM design, the FM engineer needs to consider the effectiveness of the approach in the FM design, limitations on it, and the mechanism(s) controlling the redundancy as part of the justification of the design.

Hardware identical redundancy can be used for failure detection, fault isolation, and for failure response (mitigation). A voting mechanism in a multiply-redundant computing system (usually 3 or more units) is both a mechanism for detecting failures in one of the computers and a mechanism for isolating the location of the originating fault. Hardware-identical redundancy can also be used to mitigate random failures and expected lifetime

limitations. In these cases, a single redundant unit is normally powered off until switchover when the primary unit fails. Typical examples would be the inclusion of more identical reaction wheel assemblies than required for operations, or inclusion of an identical backup computer. However, hardware-identical redundancy cannot usually mitigate a "common cause failure," a design flaw or manufacturing/assembly flaw common among all of the redundant hardware.

Functional redundancy, the use of dissimilar hardware, software, or operations procedures to perform identical functions, can be used for failure detection, by using non-identical measurements of related physical parameters. The dissimilar mechanism can provide the same information content as a crosscheck on the validity of an individual measurement. It also can be used for failure prevention, by using multiple independent mechanisms for initiating critical activities (e.g., a database enable/disable flags, an operator confirmation, and separate hardware commands to arm and fire a pyro valve). Finally, it also can be used as part of a planned autonomous failure response (e.g., failover to a "safe mode" computer) or an unplanned workaround for an in-flight anomaly (e.g., use of a thruster to replace the function of a failed reaction wheel).

Information redundancy utilizes extra information to detect, isolate, and respond to certain types of failures. The most common example is error detection and correction codes (EDAC) in which extra bits are added to a message that can be used to reconstruct the original message if some phenomenon (e.g., a single event upset (SEU)) causes one or more bits to flip.

Temporal redundancy refers to the practice of repeating a function should it fail upon a single execution. A typical example is the use of several measurements over time of the same state variable, because any single measurement could be corrupted by a SEU. Another common example in computer processing is the checkpoint-rollback capability, where a computer state is reverted (rolled back) to a previously stored computer state (the checkpoint), and then re-started to recompute the original set of calculations.

A high-cost, low risk mission may utilize several forms of redundancy to protect a required function during a critical activity. For example, a flagship robotic mission may use temporal redundancy (persistence) to avoid reacting to a transient condition, failover to a "hot" backup computer, and finally failover to a dissimilar computer, to protect operations during orbit insertion at the mission target. However, there are mission scenarios for which hardware or onboard functional or informational redundancy is not a practical option. Small, low cost missions may not be able to afford backup hardware; highly mass-constrained missions may not have the mass margins that permit multiple versions of identical hardware. In these cases, other approaches may be necessary. Software algorithms rather than alternate hardware can be used to provide functional redundancy; when time permits, operations personnel rather than onboard voting schemes can provide informational redundancy. Mission attributes, such as mission class, presence of crew, operational scenarios and specific operational hazards, mission cost/resource availability, and mission risk posture, should be used to drive the need for, and the use of, the appropriate type of redundancy.

IV. Conclusion

The principles in the NASA FM Handbook are designed to guide FM throughout the project life cycle. As such, they address both organizational and architecture/design considerations for FM. They are not intended to be a set of rules or requirements, but to provide guidance to each implementing organization within the NASA community. They represent the collected experience from decades of successful mission development and operation at NASA. They underpin and motivate the best practices identified in the NASA FM Handbook, and provide the framework within which an individual organization can define their own institutional processes and procedures.

Given that the principles are expected to represent the collected experience of FM, the authors solicit and encourage continued discussion of FM principles, best practices, and lessons learned through the ongoing review of the NASA FM Handbook, future NASA FM Workshops, and other venues, as available.

Appendix A. Glossary

Anomaly: The unexpected performance of intended function.
Failure: The unacceptable performance of an intended function.
Failure Detection: Determining that something unexpected occurred. Also referred to as fault detection.
Failure Preclusion: Actively preventing a failure from occurring.
Failure Response: An action taken to attempt to retain or regain the system's ability to control the system state in reaction to a failure.

Failure Tolerance: The ability to perform a function in the presence of any of a specified number of coincident, independent failure causes of specified types.
Fault: A physical or logical cause, which explains a failure.
Fault Management: The engineering discipline that encompasses practices which enable an operational system to contain, prevent, detect, isolate, diagnose, respond to, and recover from conditions that may interfere with nominal mission operations.
Goal Change: An action that alters the system's current objective.
Nominal: An intended, acceptable state or behavior.

Acknowledgments

Development of the Fault Management Handbook was a team effort; the full list of team members is listed in the FM Handbook. The authors of this paper specifically thank M. Clark, J. Day, N. Dennehy, P. Hattis, S. Johnson, D. McComas, E. Rice, J. West and J. Zinchuck for their contributions to and review of the FM Principles presented in this paper.

Part of this research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1]Barley, B., Gilbert, P., and Newhouse, M., "Improving the Life Cycle Cost Management of Planetary Missions," February 2010.
[2]Fesq, L., Cancro, G. Jones, C., Ingham, M., Leitner, J., McDougal, J., Newhouse, M., Rice, E., Watson, D., Wertz, J., "Spacecraft Fault Management Workshop Results for the Science Mission Directorate, Planetary Sciences Division," March 2009.
[3]*NASA Fault Management Handbook,* Preliminary, Washington, DC, 2011 (to be published).
[4]NASA/SP-2007-6105 Rev. 1, Systems Engineering Handbook. Washington, DC, 2007.
[5]Columbia Accident Investigation Board, *Columbia Accident Investigation Board Report,* Vol.1, Washington, DC, 2003.